

Introduction to Data Science – Week 5

HEZI RESHEFF

COURSE 55793

2022-2023



Lessons 1-4 recap

- data types: int, float, bool, list, set, tuple, dict
- if statements
- iterators and iterability
- loops: for, while
- functions



A note about the exercise

- strings are iterable.

```
In [1]: s = "abcdefg"
```

```
In [2]: list(s)
```

```
Out[2]: ['a', 'b', 'c', 'd', 'e', 'f', 'g']
```

```
In [3]: for char in s:  
        print(char)
```

```
a  
b  
c  
d  
e  
f  
g
```

This week

- Finishing up dict (sorting by key or value...)
- Reading and writing files
- Introduction to the **pandas** library



Install package from jupyter notebook

In [54]: `!pip install wikipedia`

```
Collecting wikipedia
  Downloading wikipedia-1.4.0.tar.gz (27 kB)
Requirement already satisfied: beautifulsoup4 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from wikipedia) (4.11.1)
Requirement already satisfied: requests<3.0.0,>=2.0.0 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from wikipedia) (2.27.1)
Requirement already satisfied: idna<4,>=2.5 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from requests<3.0.0,>=2.0.0->wikipedia) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from requests<3.0.0,>=2.0.0->wikipedia) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from requests<3.0.0,>=2.0.0->wikipedia) (1.26.9)
Requirement already satisfied: certifi>=2017.4.17 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from requests<3.0.0,>=2.0.0->wikipedia) (2021.10.8)
Requirement already satisfied: soupsieve>1.2 in /Users/zeekresheff/opt/anaconda3/lib/python3.9/site-packages (from beautifulsoup4->wikipedia) (2.3.1)
Building wheels for collected packages: wikipedia
  Building wheel for wikipedia (setup.py) ... done
  Created wheel for wikipedia: filename=wikipedia-1.4.0-py3-none-any.whl size=11695 sha256=6cb4177dae0a6687b76396f2a6d9f35300bb3d9f01ae9adf5ef37e7b9bb45e39
  Stored in directory: /Users/zeekresheff/Library/Caches/pip/wheels/c2/46/f4/caa1bee71096d7b0cdca2f2a2af45cacf35c5760bee8f00948
Successfully built wikipedia
Installing collected packages: wikipedia
Successfully installed wikipedia-1.4.0
```



Get text from Wikipedia

```
In [55]: import wikipedia
```

```
In [57]: wiki = wikipedia.page('Python (programming language)')
```

```
In [58]: wiki.content
```

```
Out[58]: 'Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages.\n\n\n== History ==\n\nPython was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC programming language, which was inspired by SETL, capable of exception handling (from the start plus new capabilities in Python 3.11) and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's "benevolent dictator for life", a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a five-member Steering Council to lead the project. Python 2.0 was released on 16 October 2000, with many major new features. Python 3.0, released on 3 December 2008, with many of its major features backported to Python 2.6.x and 2.7.x. Releases of Python 3 include the 2to3 utility, which automates the translation of Python 2 code to Python 3. Python 2.7's end-of-life was initially set for 2015, but was extended to 2020 due to a number of factors, including the need for security updates and the fact that Python 2 was still the most widely used version of the language. Python 3.12 is the current stable release, and Python 3.13 is expected to be released in late 2024. Python 3.14 is expected to be released in late 2025. Python 3.15 is expected to be released in late 2026. Python 3.16 is expected to be released in late 2027. Python 3.17 is expected to be released in late 2028. Python 3.18 is expected to be released in late 2029. Python 3.19 is expected to be released in late 2030. Python 3.20 is expected to be released in late 2031. Python 3.21 is expected to be released in late 2032. Python 3.22 is expected to be released in late 2033. Python 3.23 is expected to be released in late 2034. Python 3.24 is expected to be released in late 2035. Python 3.25 is expected to be released in late 2036. Python 3.26 is expected to be released in late 2037. Python 3.27 is expected to be released in late 2038. Python 3.28 is expected to be released in late 2039. Python 3.29 is expected to be released in late 2040. Python 3.30 is expected to be released in late 2041. Python 3.31 is expected to be released in late 2042. Python 3.32 is expected to be released in late 2043. Python 3.33 is expected to be released in late 2044. Python 3.34 is expected to be released in late 2045. Python 3.35 is expected to be released in late 2046. Python 3.36 is expected to be released in late 2047. Python 3.37 is expected to be released in late 2048. Python 3.38 is expected to be released in late 2049. Python 3.39 is expected to be released in late 2050. Python 3.40 is expected to be released in late 2051. Python 3.41 is expected to be released in late 2052. Python 3.42 is expected to be released in late 2053. Python 3.43 is expected to be released in late 2054. Python 3.44 is expected to be released in late 2055. Python 3.45 is expected to be released in late 2056. Python 3.46 is expected to be released in late 2057. Python 3.47 is expected to be released in late 2058. Python 3.48 is expected to be released in late 2059. Python 3.49 is expected to be released in late 2060. Python 3.50 is expected to be released in late 2061. Python 3.51 is expected to be released in late 2062. Python 3.52 is expected to be released in late 2063. Python 3.53 is expected to be released in late 2064. Python 3.54 is expected to be released in late 2065. Python 3.55 is expected to be released in late 2066. Python 3.56 is expected to be released in late 2067. Python 3.57 is expected to be released in late 2068. Python 3.58 is expected to be released in late 2069. Python 3.59 is expected to be released in late 2070. Python 3.60 is expected to be released in late 2071. Python 3.61 is expected to be released in late 2072. Python 3.62 is expected to be released in late 2073. Python 3.63 is expected to be released in late 2074. Python 3.64 is expected to be released in late 2075. Python 3.65 is expected to be released in late 2076. Python 3.66 is expected to be released in late 2077. Python 3.67 is expected to be released in late 2078. Python 3.68 is expected to be released in late 2079. Python 3.69 is expected to be released in late 2080. Python 3.70 is expected to be released in late 2081. Python 3.71 is expected to be released in late 2082. Python 3.72 is expected to be released in late 2083. Python 3.73 is expected to be released in late 2084. Python 3.74 is expected to be released in late 2085. Python 3.75 is expected to be released in late 2086. Python 3.76 is expected to be released in late 2087. Python 3.77 is expected to be released in late 2088. Python 3.78 is expected to be released in late 2089. Python 3.79 is expected to be released in late 2090. Python 3.80 is expected to be released in late 2091. Python 3.81 is expected to be released in late 2092. Python 3.82 is expected to be released in late 2093. Python 3.83 is expected to be released in late 2094. Python 3.84 is expected to be released in late 2095. Python 3.85 is expected to be released in late 2096. Python 3.86 is expected to be released in late 2097. Python 3.87 is expected to be released in late 2098. Python 3.88 is expected to be released in late 2099. Python 3.89 is expected to be released in late 2100. Python 3.90 is expected to be released in late 2101. Python 3.91 is expected to be released in late 2102. Python 3.92 is expected to be released in late 2103. Python 3.93 is expected to be released in late 2104. Python 3.94 is expected to be released in late 2105. Python 3.95 is expected to be released in late 2106. Python 3.96 is expected to be released in late 2107. Python 3.97 is expected to be released in late 2108. Python 3.98 is expected to be released in late 2109. Python 3.99 is expected to be released in late 2110. Python 4.0 is expected to be released in late 2111. Python 4.1 is expected to be released in late 2112. Python 4.2 is expected to be released in late 2113. Python 4.3 is expected to be released in late 2114. Python 4.4 is expected to be released in late 2115. Python 4.5 is expected to be released in late 2116. Python 4.6 is expected to be released in late 2117. Python 4.7 is expected to be released in late 2118. Python 4.8 is expected to be released in late 2119. Python 4.9 is expected to be released in late 2120. Python 4.10 is expected to be released in late 2121. Python 4.11 is expected to be released in late 2122. Python 4.12 is expected to be released in late 2123. Python 4.13 is expected to be released in late 2124. Python 4.14 is expected to be released in late 2125. Python 4.15 is expected to be released in late 2126. Python 4.16 is expected to be released in late 2127. Python 4.17 is expected to be released in late 2128. Python 4.18 is expected to be released in late 2129. Python 4.19 is expected to be released in late 2130. Python 4.20 is expected to be released in late 2131. Python 4.21 is expected to be released in late 2132. Python 4.22 is expected to be released in late 2133. Python 4.23 is expected to be released in late 2134. Python 4.24 is expected to be released in late 2135. Python 4.25 is expected to be released in late 2136. Python 4.26 is expected to be released in late 2137. Python 4.27 is expected to be released in late 2138. Python 4.28 is expected to be released in late 2139. Python 4.29 is expected to be released in late 2140. Python 4.30 is expected to be released in late 2141. Python 4.31 is expected to be released in late 2142. Python 4.32 is expected to be released in late 2143. Python 4.33 is expected to be released in late 2144. Python 4.34 is expected to be released in late 2145. Python 4.35 is expected to be released in late 2146. Python 4.36 is expected to be released in late 2147. Python 4.37 is expected to be released in late 2148. Python 4.38 is expected to be released in late 2149. Python 4.39 is expected to be released in late 2150. Python 4.40 is expected to be released in late 2151. Python 4.41 is expected to be released in late 2152. Python 4.42 is expected to be released in late 2153. Python 4.43 is expected to be released in late 2154. Python 4.44 is expected to be released in late 2155. Python 4.45 is expected to be released in late 2156. Python 4.46 is expected to be released in late 2157. Python 4.47 is expected to be released in late 2158. Python 4.48 is expected to be released in late 2159. Python 4.49 is expected to be released in late 2160. Python 4.50 is expected to be released in late 2161. Python 4.51 is expected to be released in late 2162. Python 4.52 is expected to be released in late 2163. Python 4.53 is expected to be released in late 2164. Python 4.54 is expected to be released in late 2165. Python 4.55 is expected to be released in late 2166. Python 4.56 is expected to be released in late 2167. Python 4.57 is expected to be released in late 2168. Python 4.58 is expected to be released in late 2169. Python 4.59 is expected to be released in late 2170. Python 4.60 is expected to be released in late 2171. Python 4.61 is expected to be released in late 2172. Python 4.62 is expected to be released in late 2173. Python 4.63 is expected to be released in late 2174. Python 4.64 is expected to be released in late 2175. Python 4.65 is expected to be released in late 2176. Python 4.66 is expected to be released in late 2177. Python 4.67 is expected to be released in late 2178. Python 4.68 is expected to be released in late 2179. Python 4.69 is expected to be released in late 2180. Python 4.70 is expected to be released in late 2181. Python 4.71 is expected to be released in late 2182. Python 4.72 is expected to be released in late 2183. Python 4.73 is expected to be released in late 2184. Python 4.74 is expected to be released in late 2185. Python 4.75 is expected to be released in late 2186. Python 4.76 is expected to be released in late 2187. Python 4.77 is expected to be released in late 2188. Python 4.78 is expected to be released in late 2189. Python 4.79 is expected to be released in late 2190. Python 4.80 is expected to be released in late 2191. Python 4.81 is expected to be released in late 2192. Python 4.82 is expected to be released in late 2193. Python 4.83 is expected to be released in late 2194. Python 4.84 is expected to be released in late 2195. Python 4.85 is expected to be released in late 2196. Python 4.86 is expected to be released in late 2197. Python 4.87 is expected to be released in late 2198. Python 4.88 is expected to be released in late 2199. Python 4.89 is expected to be released in late 2200. Python 4.90 is expected to be released in late 2201. Python 4.91 is expected to be released in late 2202. Python 4.92 is expected to be released in late 2203. Python 4.93 is expected to be released in late 2204. Python 4.94 is expected to be released in late 2205. Python 4.95 is expected to be released in late 2206. Python 4.96 is expected to be released in late 2207. Python 4.97 is expected to be released in late 2208. Python 4.98 is expected to be released in
```



Let's count words!

```
In [61]: count_dict = {}  
  
        for word in wiki.content.split(" "):  
            count_dict.setdefault(word, 0)  
            count_dict[word] = count_dict[word] + 1
```

```
In [68]: count_dict["Python"]
```

```
Out[68]: 134
```

```
In [69]: count_dict["Python."]
```

```
Out[69]: 6
```

```
In [70]: count_dict["Python,"]
```

```
Out[70]: 10
```



Class exercise: count without the punctuation

- Use string functions (such as `.replace`) to count the number of words regardless of:
 - Punctuation before/after the word
 - Upper or lower case



Sorting by key

- To sort a dictionary by key (words) we use:

```
In [75]: sorted(count_dict.items())
```

```
('CPython', 8),  
('CPython)', 1),  
('CPython.', 1),  
('Canopy', 1),  
('Centrum', 1),  
('Child', 1),  
('Cinema', 1),  
('CircuitPython', 1),  
('Circus.', 1),  
('Class.method(instance,', 1),  
('Code', 1),  
('Comedy', 1),  
('Common', 2),  
('Community', 1),  
('Computer', 2),  
('Computerworld.', 1),  
('Council', 1),  
('Cross-compilers', 1),  
('Currently', 1),  
('Cython', 1)
```



Sorting by value

- To sort a dictionary by value (word count) we need a more complicated functionality: a key function
- a mapping: object -> number

```
In [76]: def key_func(item):  
         key, value = item  
         return value
```

```
In [79]: sorted(count_dict.items(), key=key_func, reverse=True)
```

```
Out[79]: [('the', 199),  
          ('and', 188),  
          ('a', 161),  
          ('to', 147),  
          ('Python', 134),  
          ('of', 122),  
          ('in', 111),  
          ('is', 105),  
          ('as', 70),  
          ('for', 62),  
          ('are', 49),  
          ('with', 47),  
          ('be', 42),  
          ('that', 36),  
          ('by', 34),
```



Looping through Containers: Enumerate

The built-in function `enumerate(seq)` returns the position index and corresponding value at the same time.

Example:

```
In [80]: top10 = sorted(count_dict.items(), key=key_func, reverse=True)[:10]
```

```
In [90]: for i, item in enumerate(top10):  
         word, num = item  
         print(f"{i}: the word {word.upper()} appears {num} times.")
```

```
0: the word THE appears 199 times.  
1: the word AND appears 188 times.  
2: the word A appears 161 times.  
3: the word TO appears 147 times.  
4: the word PYTHON appears 134 times.  
5: the word OF appears 122 times.  
6: the word IN appears 111 times.  
7: the word IS appears 105 times.  
8: the word AS appears 70 times.  
9: the word FOR appears 62 times.
```



Exercise

- Enumrate all the words that appear at least 25 times



A solution

```
In [91]: for i, item in enumerate(sorted(count_dict.items(), key=key_func, reverse=True)):
        word, num = item
        if num < 25:
            break
        print(f"{i}: the word {word.upper()} appears {num} times.")
```

```
0: the word THE appears 199 times.
1: the word AND appears 188 times.
2: the word A appears 161 times.
3: the word TO appears 147 times.
4: the word PYTHON appears 134 times.
5: the word OF appears 122 times.
6: the word IN appears 111 times.
7: the word IS appears 105 times.
8: the word AS appears 70 times.
9: the word FOR appears 62 times.
10: the word ARE appears 49 times.
11: the word WITH appears 47 times.
12: the word BE appears 42 times.
13: the word THAT appears 36 times.
14: the word BY appears 34 times.
15: the word USED appears 31 times.
16: the word WHICH appears 30 times.
17: the word OR appears 29 times.
18: the word HAS appears 29 times.
19: the word FROM appears 28 times.
20: the word PROGRAMMING appears 26 times.
21: the word WAS appears 26 times.
22: the word ALSO appears 25 times.
```



Looping through Containers: Zip

Make an iterator that aggregates elements from each of the iterables. Returns an iterator of tuples, where the i-th tuple contains the i-th element from each of the argument sequences or iterables. The iterator stops when the shortest input iterable is exhausted.

```
In [147]: x = [1,2,3]
In [148]: y = [4,5,6]
In [149]: zip(x,y)
Out[149]: <zip at 0x111811f88>
```

```
In [150]: list(_)
Out[150]: [(1, 4), (2, 5), (3, 6)]
```

```
In [151]: questions = ['name', 'quest', 'favorite color']
In [152]: answers = ['lancelot', 'the holy grail', 'blue']
```

```
In [153]: for q, a in zip(questions, answers):
...:     print('What is your {0}? It is {1}'.format(q, a))
...:
```

```
What is your name? It is lancelot.
What is your quest? It is the holy grail.
What is your favorite color? It is blue.
```

Unpacking tuples



Comparing collections

type	What is it	Create	Orderd	Iterable	Mutable
tuple	Easy way to group multiple “things” in a single variable -- t = 1, 2, 3 -- t = (1,2,3) -- return a, b	a,b	Yes	Yes	No
list	Ordered collection of elements -- li = [1,2,3]	[a,b]	Yes	Yes	Yes
set	Unordered collection of elements without repeatition -- s = {1,2,3}	{a,b}	No	Yes	Yes
dict	Mapping from key to value -- d = {"a": 1, "b": 2}	{"key a": "val a"}	No	Yes	Yes (values)



01011000101010111000011001011000110011001001010001
10011010000100101110100100011011101100110000100100
10100100000100110111000010101010111001110010000101011
0101011010011000101000101010011011010010100100010
00011000001110101101100010101001011001010100100101
11100101010000100111100010100101110100001010010100
100010110110001000111010000110110110010010010100
01011010001101110000010101011001000010100111100001
01001010110001101001101100100001110000011101001001
101000001010100110011001100101010010101100001101010
01011000101010111000011001011000110011001001010001
10011010000100101110100100011011101100110000100100
1010010000010011011100001010100101110010000101011
01010110100110001010101010100110110100100100010
00011000001110101101100010101001011001010100100101
1110010101000010011100010100101110100001010010100
100010110110001000111010000110110110010010010100
01011010001101110000010101011001000010100111100001
0100101011000101001101100100001110000011101001001
1010000010101001100110011010101010100101100001101010
01000100010101110100011010010110110011000100000111
010110001010101110000100101100010011001001010001
10011010000100101110100100011011101100110000100100
1010010000010011011100001010100101110010000101011
01011000101010111000010010111000110011001001010001
10011010000101001110100100011011101100110000100100
10100100000100110110000101010101110010000101011
01011000101010111000010010111000110011001001010001
10100100000100110110000101010101110010000101011
0101011010011000101010101011011010010100100010
0001100000111010110110001010101010100101010010101
10001011011000100011010000110110110010010101000
01011010001101110000010101011001000010100111100001
01001010110001101001101100100001110000011101001001

Reading from files

```
f = open("thisisafire", "r")  
  
row1 = f.readline()  
print(row1)  
  
row2 = f.readline()  
print(row2)  
  
f.close()
```

This is text in a file

And more here ...

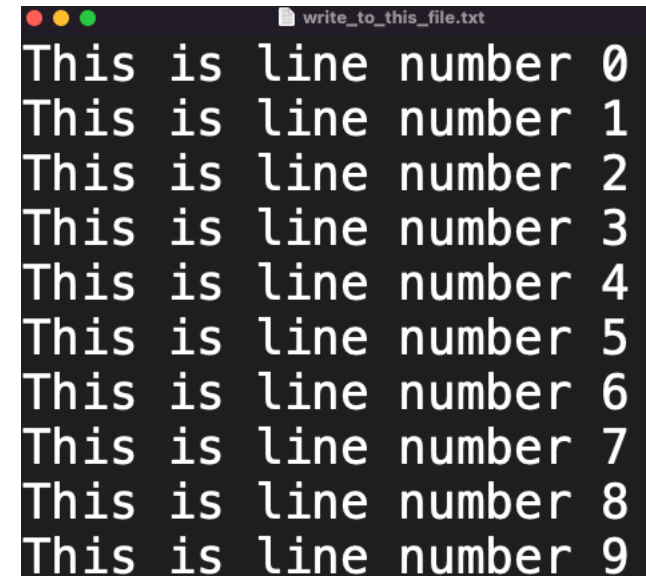


Writing to files

```
write_file = open("write_to_this_file.txt", "w")

for i in range(10):
    s = f"This is line number {i}\n"
    write_file.write(s)

write_file.close()
```



A screenshot of a text editor window titled "write_to_this_file.txt". The window displays 10 lines of text, each starting with "This is line number" followed by a number from 0 to 9. The text is white on a dark background.

```
This is line number 0
This is line number 1
This is line number 2
This is line number 3
This is line number 4
This is line number 5
This is line number 6
This is line number 7
This is line number 8
This is line number 9
```

With block

- Replace the open-close logic:

```
with open("thisisafire", "r") as f:  
    data = f.readlines()  
  
# The file is no longer open  
for row in data:  
    print(row)
```

This is text in a file

And more here ...

Exercise

- Write a function:

```
def sum_from_file(file_name):  
    ...
```

that takes the name of a file, reads the file row by row, and adds up all rows that contain a number. For example if the file contains:

```
bla bla this is not a number  
8  
3  
bla bla this is not a number either
```

Then the output is 11.

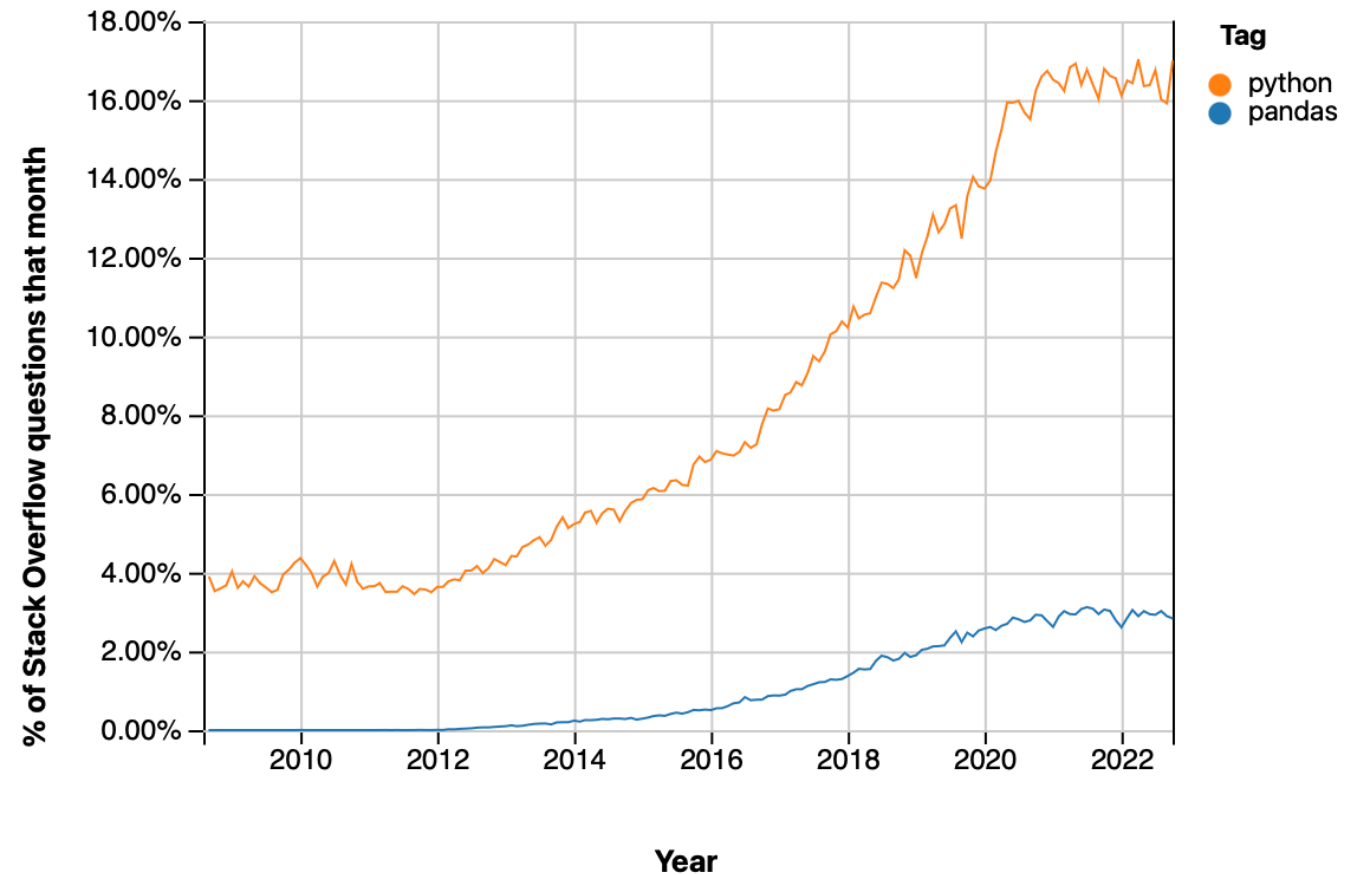
Pandas

Pandas

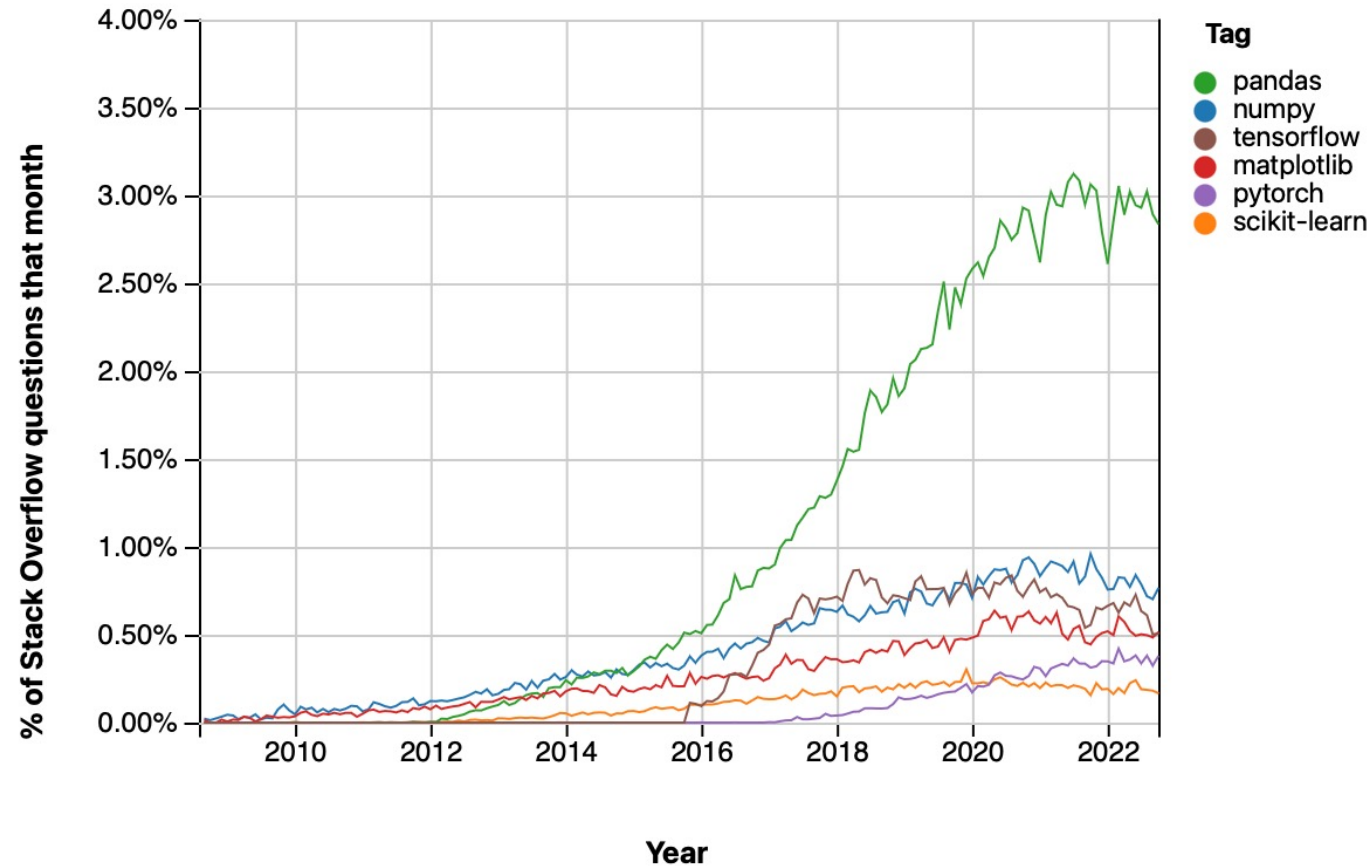
- Panel data (observations over time/ Python data analysis)
- But now much more...
- Very popular data science tool
- Tabular data



Pandas popularity on the rise



Pandas vs. other data-science packages



Pandas objects

- Series
 - Like a single excel column
- DataFrame
 - Like a single spreadsheet (each column is a Series)

Pandas Series

```
In [1]: import pandas as pd
```

```
In [2]: s = pd.Series([4, 9, 17, 0, 5], name="my series")
```

```
In [3]: s
```

```
Out[3]: 0      4  
        1      9  
        2     17  
        3      0  
        4      5  
        Name: my series, dtype: int64
```

type, dtype, name

```
In [4]: type(s)
```

```
Out[4]: pandas.core.series.Series
```

```
In [5]: s.dtype
```

```
Out[5]: dtype('int64')
```

```
In [6]: s.name
```

```
Out[6]: 'my series'
```

Series indexing

```
s2 = pd.Series([1, 2, 3, 4], index=["a", "b", "c", "d"])
```

```
s2
```

```
a    1  
b    2  
c    3  
d    4  
dtype: int64
```

```
In [21]: s2["a"]
```

```
Out[21]: 1
```

```
In [22]: s2[0]
```

```
Out[22]: 1
```

```
In [23]: s2.iloc[0:2]
```

```
Out[23]: a    1  
         b    2  
         dtype: int64
```

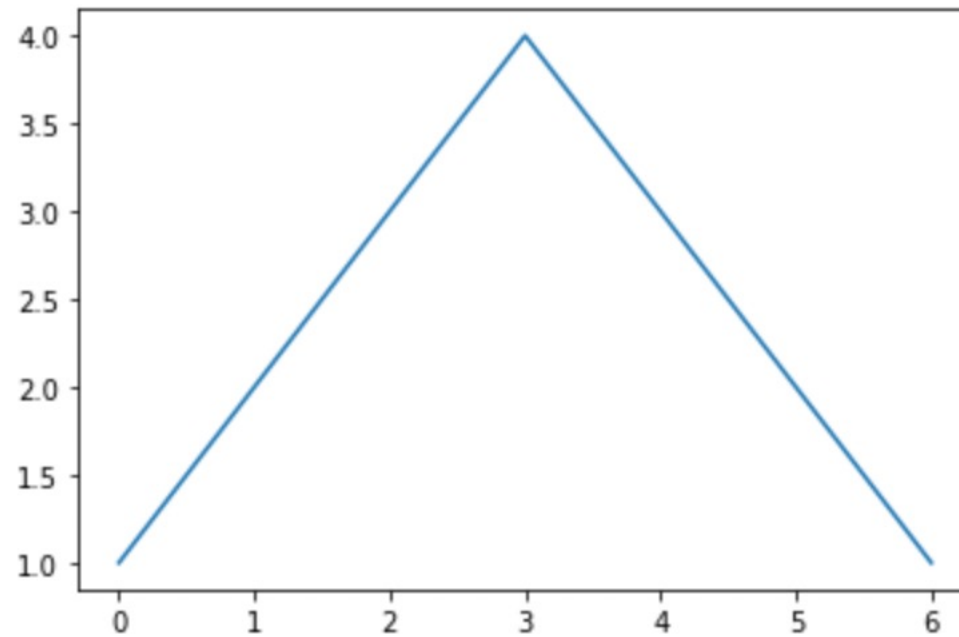
```
In [24]: s2.loc[:"c"]
```

```
Out[24]: a    1  
         b    2  
         c    3  
         dtype: int64
```

Plot

```
s3 = pd.Series([1, 2, 3, 4, 3, 2, 1])  
s3.plot()
```

<AxesSubplot:>



Pandas DataFrame

```
data = {"col1": [1,2,3], "col2": [4,5,6]}  
f = pd.DataFrame(data)  
f
```

	col1	col2
0	1	4
1	2	5
2	3	6

```
data = [[1,2,3],  
        [4,5,6]]  
f2 = pd.DataFrame(data,  
                   columns=["col1","col2","col3"],  
                   index=["row1", "row2"])  
f2
```

	col1	col2	col3
row1	1	2	3
row2	4	5	6

DataFrame from multiple Series

```
s1 = pd.Series([1,2,3,4])
s2 = pd.Series(["hello", "these", "are", "words"])

f = pd.DataFrame({"numeric-column": s1, "string-column": s2})

f
```

	numeric-column	string-column
0	1	hello
1	2	these
2	3	are
3	4	words

info

```
f.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 4 entries, 0 to 3
```

```
Data columns (total 2 columns):
```

#	Column	Non-Null Count	Dtype
0	numeric-column	4 non-null	int64
1	string-column	4 non-null	object

```
dtypes: int64(1), object(1)
```

```
memory usage: 192.0+ bytes
```


DataFrame column and index

```
f.index = ["a", "b", "c", "d"]  
f
```

	numeric-column	string-column
a	1	hello
b	2	these
c	3	are
d	4	words

```
f.columns = ["col1", "col2"]  
f
```

	col1	col2
a	1	hello
b	2	these
c	3	are
d	4	words

DataFrame indexing

```
f["col1"]
```

```
a    1  
b    2  
c    3  
d    4  
Name: col1, dtype: int64
```

get column (Series)



```
f["a":"c"]
```

	col1	col2
a	1	hello
b	2	these
c	3	are

rows by index



```
f.loc["a": "c"]
```

	col1	col2
a	1	hello
b	2	these
c	3	are

rows by index



```
f.iloc[:3]
```

	col1	col2
a	1	hello
b	2	these
c	3	are

rows by position



```
f.loc["b", "col2"]
```

'these'

```
f.iloc[1, 1]
```

'these'

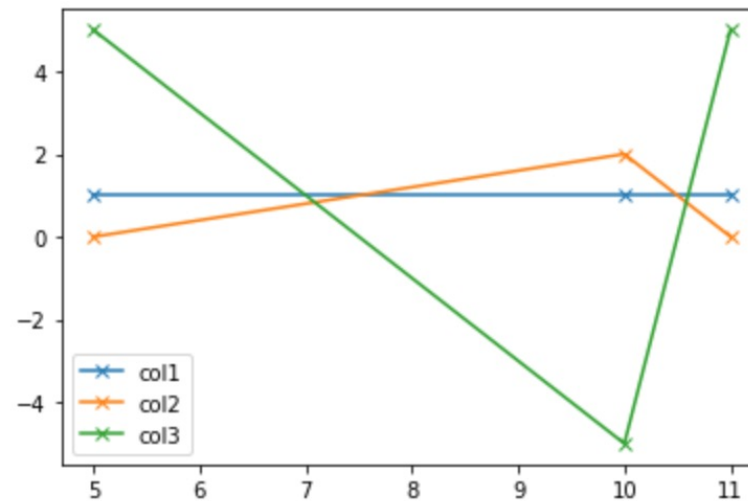
cell by index/column



Plot

```
data = [[1, 0, 5],  
        [1, 2, -5],  
        [1, 0, 5]]  
  
f = pd.DataFrame(data,  
                  columns=["col1", "col2", "col3"],  
                  index=[5, 10, 11])  
  
f.plot(style="-x")
```

<AxesSubplot:>



read_csv

```
netflix_data = pd.read_csv("~/Desktop/netflix_titles.csv")
```

```
netflix_data.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train l...

Make the title column the index

```
netflix_data.set_index("title", inplace=True)
```

```
netflix_data.head()
```

	show_id	type	director	cast	country	date_added	release_year	rating	duration	listed_in	description
title											
Dick Johnson Is Dead	s1	Movie	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
Blood & Water	s2	TV Show	NaN	Ama Qamata, Khosi Ngema, Gail Mabalan...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
Ganglands	s3	TV Show	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...
Jailbirds New Orleans	s4	TV Show	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...
Kota Factory	s5	TV Show	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train l...

Class exercise

- load the data using `rad_csv`
- make the movie title the index using `set_index`
- print the description field of "The Matrix"

Filtering a DataFrame

- What if I only want rows of Movies (no TV)?

```
filter_by = netflix_data["type"] == "Movie"
netflix_movies_only = netflix_data.loc[filter_by]
```

```
netflix_movies_only.sample(3)
```

	show_id	type	director	cast	country	date_added	release_year	rating	duration	listed_in	description
title											
God Calling	s1108	Movie	BB Sasore	Zainab Balogun, Karibi Fubara, Diana Egwuatu, ...	Nigeria	April 2, 2021	2018	TV-MA	120 min	Dramas, Faith & Spirituality, International Movies	The loss of a child plunges Sade into a suicid...
My Happy Family	s5144	Movie	Nana Ekvimishvili, Simon Gross	Ia Shugliashvili, Merab Ninidze, Berta Khapava...	Georgia, Germany, France	December 1, 2017	2017	TV-14	119 min	Dramas, Independent Movies, International Movies	A middle-aged wife and mother of two shocks he...
Katti Batti	s4449	Movie	Nikhil Advani	Imran Khan, Kangana Ranaut, Abhishek Saha, Mit...	India	November 1, 2018	2015	TV-14	132 min	Comedies, Dramas, International Movies	After falling in love during college, architec...

Filter rows and select columns

```
filter_rows_by = netflix_data["country"] == "Israel"
keep_columns = ["date_added", "duration", "description"]

netflix_data.loc[filter_rows_by, keep_columns]
```

	date_added	duration	description
title			
Shtisel	July 6, 2021	3 Seasons	A Haredi family living in an ultra-Orthodox ne...
Black Space	May 27, 2021	1 Season	A rogue detective with unorthodox means leads ...
Fauda	April 16, 2020	3 Seasons	A top Israeli agent comes out of retirement to...
When Heroes Fly	January 10, 2019	1 Season	Years after a bitter falling out, four Israeli...
Hashoter Hatov	December 28, 2018	1 Season	An honest – though overzealous – police office...
Mossad 101	July 11, 2018	2 Seasons	Cadets from every level of Israeli society und...
Maktub	June 15, 2018	106 min	After surviving a bomb attack, two low-level m...
Inside the Mossad	January 29, 2019	1 Season	In this documentary, dozens of former agents f...
Numbered	December 31, 2017	55 min	Guided by survivors' testimonies, this documen...
Shadow of Truth	January 27, 2017	1 Season	This documentary series explores the explosive...
Suicide (Hitabdut)	July 1, 2016	114 min	A failed businessman must kill himself to pay ...
The Golem	June 26, 2019	95 min	As sickness spreads throughout the countryside...
The Women's Balcony	December 19, 2017	100 min	A conservative rabbi steps in to lead a congre...

Class exercise

- Find all the TV shows from Finland
- Print 2 rows of Icelandic TV shows or movies
- In which year were more titles released, 2019 or 2020?

Adding columns to a DataFrame

- Why add columns? Like in Excel... as part of a computation, as output

```
netflix_data["new-col"] = 0
```

```
netflix_data["year_before"] = netflix_data["release_year"] - 1
```

save DataFrame as csv

- `my_frame.to_csv(path)`

Exercise

- Write code that goes over the netflix data dataframe, and prints only movies where the description contains the name of the movie.
- What if we wanted to count them instead of print?